Enabling Real-Time Wireless Channel Based Encryption Key Generation

A Thesis

Submitted to the Faculty

of

Drexel University

by

Brandon Z. Katz

in partial fulfillment of the

requirements for the degree

of

Master of Science in Electrical Engineering

May 2016

DEDICATION

To the adventure.

ACKNOWLEDGEMENTS

I must express my sincere gratitude to the many people who helped me along through this endeavor. My advisor, Kapil Dandekar, for introducing me to physical layer security and providing me the opportunity to pursue my interests under his guidance in his research group. My committee members, Steven Weber and Geoffrey Mainland for giving up their time to review my thesis and defense and providing me with critical feedback. Cem Sahin for helping me develop a meaningful understanding of channel based encryption key generation and discussing the nuances of theoretical algorithm implementation. Simon Begashaw for his willingness to always discuss the fundamental concepts behind my work to help improve my understanding and in turn improve my designs. Danh Nguyen for helping me work with the software reference designs I chose to implement my system with. James Chacko for patiently showing me how to work with hardware reference designs. Marko Jacovic for the advice and hardware modification help. Ilhaan Rasheed and Darshan Donthi for their understanding when trying to get class work finished. Oday Bshara for his words of wisdom and for making the lab feel a little less lonely late at night. All of the members of the Drexel Wireless Systems Laboratory for welcoming me into their group and offering me the same support as they offer each other. My friends and colleagues at

LM-ATL for introducing me to wireless systems research. Jenna Schabdach and Dana Bronen for the thesis support and making long nights of writing much more bearable. My family, Michael, Paula, Emily, Miranda, and Szasz for their unending support and encouragement. Bob, Eric, and Stephen, for making sure our house stayed standing and understanding my absence. And last but perhaps most important, all of the people that I am so fortunate to be able to know as my friends for their part in this adventure.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Enabling Real-Time Wireless Channel Based Encryption Key Generation

Brandon Z. Katz

Advisor: Kapil R. Dandekar, Ph.D.

As the demand for wireless communication grows, so does the need for wireless security. Through both high profile attacks as well as personal identity theft at open access points, it has been demonstrated that security is falling behind the curve. Wireless consumers create weak passwords, forget to turn on the latest encryption, and connect to open wireless networks every day. One solution is to remove the human element and instead generate encryption keys on the fly. Recently, methods have been proposed to use wireless channels as common entropy sources to enable radios to generate symmetric encryption keys. These techniques rely on the unique wireless environment between two radios in order to effectively move security down to the physical layer of a radio network as opposed to requiring users to handle encryption keys directly. In this thesis, a method of generating keys using wireless channels in real-time through additions to the IEEE 802.11 protocol is described and validated. As part of the process, a technique for sampling wireless channels using application layer traffic is designed and tested. Additionally, major related points of interest such as channel coherence, bit error handling, and real-time processing, are discussed.

CHAPTER 1: INTRODUCTION

The ever increasing role that wireless devices play in everyday life allows information to flow faster than ever. From messages to loved ones, sensitive banking information, and critical medical monitoring, much personal information travels through the air at some point on its journey. While this wireless revolution has added daily convenience and improved overall quality of life for millions of people, it has also created new vulnerabilities that were unheard of just twenty years ago.

Wireless security always seems to be one step behind wireless intruders. Many networks are completely open, and outdated encryption techniques are still commonly used. These deficiencies allow the theft of personal data and make illegal eavesdropping quick and easy to implement.

One major commonality between existing encryption schemes is the use of pre-shared secret keys or the use of a public key system. While pre-shared keys can be strengthened through the use of longer and more random keys, they are susceptible to man-in-the-middle and eavesdropping-style attacks. In common home network security schemes, where the user has input to the key generation process, networks also become vulnerable to simple brute-force-style attacks due to weak passphrases. Still, despite our best data security efforts, we also often encounter unsecured wireless networks in public places.

One potential solution to the stated attacks, and even the public network problem, is to

move the encryption key life-cycle down to the physical layer of communications networks from key generation and distribution through data encryption. This research began in the early 1990s with investigations into extracting mutual secret information from correlated random variables. As wireless channels are naturally random (yet can be correlated under certain circumstances) [1], some information theoretic studies turned towards using wireless channels between two radios as a shared entropy source. By the early 2000s, researchers were looking at the experimental aspects of channel based encryption key generation using very controlled testbeds, and as a result the popular 'Radio-Telepathy' algorithm was introduced [2]. While new algorithms were developed to extract bits from wireless channels [3], very little progress was made on transitioning the technology towards widely adopted systems.

In this thesis, one facet of physical layer security, channel based symmetric encryption key generation, will be investigated. Specifically, this thesis aims to:

- Design, construct, and verify a proof of concept real-time channel based key generation system
- Formulate a technique to sample wireless channels using existing information from application layer communications building on work presented in [4].

In order to accomplish these goals, first, a background of wireless security techniques and foundations of Physical (PHY) layer key generation will be given in chapter 2. Next, a novel technique for sampling wireless channels with minimal overhead will be developed in chapter 3. The discussed sampling technique will then be incorporated into one of the first real-time PHY layer key generation systems in chapter 4. Chapter 5 will cover how the sampling technique and real-time system were implemented for experimentation while

results are presented along with discussion in chapter 6. Finally, conclusions will be drawn and recommendations for future work put forward in chapter 7.

CHAPTER 2: BACKGROUND

In this chapter, an introduction to wireless security will be provided following the evolution of the IEEE 802.11 standard. Next, potential existing solutions to deficiencies will be brought to light. The concept of physical layer security will be presented, followed by a brief discussion of wireless channels. Finally, the idea of using wireless channels to generate encryption keys will be presented.

2.1 Modern Wireless Security

The modern exchange of information over wireless networks poses security challenges never dreamt of by early wireless pioneers. As the IEEE 802.11 standard is used as a test system throughout this thesis, a brief overview of 802.11 security mechanisms is provided here. Wired Equivalent Privacy (WEP) was the security standard introduced with 802.11 in 1997 [5]. WEP used the Rivest Cipher 4 (RC4) stream cipher for encryption key generation which was simply XORed with plaintext. While an Initialization Vector (IV) was included in WEP to keep the cipher from repeating, weak IV generation undermined much of the effort. Overall, WEP was broken within four years of its introduction [6]. Multiple research groups have shown that WEP keys can be discovered through capturing over the air traffic for less than one minute. Once known, the key can be used to decrypt all traffic. Even today, WEP security continues to plague 802.11 networks as many legacy systems still use

it as their primary security method.

In 2004, WEP was deprecated with the 802.11i amendment [7]. In its place, WiFi Protected Access (WPA) was introduced as a more secure option than WEP, which could be applied with a software upgrade to existing hardware. While WPA still used the RC4 stream cipher, it improved encryption through the use of what is known as the Temporal Key Integrity Protocol (TKIP). TKIP helps to randomize keys through the use of a hashing function before using them to seed RC4, allowing each packet to be encrypted with a unique key. Despite the improvements offered by WPA with TKIP, keys were easily broken when users used short passphrases. In addition, TKIP is vulnerable to packet insertion through a time consuming, but usable, attack [8], [9].

In order to remove the weakness caused by the RC4 cipher, the 802.11i amendment introduced the use of the Advanced Encryption Standard (AES) block cipher. This part of the amendment, called WiFi Protected Access Second Generation (WPA2) by the industry, required hardware changes and was meant to complete the security overhaul started with WPA-TKIP. The use of AES instead of RC4 makes WPA2 a much stronger security mechanism than previous solutions, and as a result WPA2 is now the only non-deprecated 802.11 security mechanism as of 2012 [10]. WPA2 can operate in a 'personal' mode which utilizes a Pre-Shared Key (PSK) between the Access Point (AP) and each Station (STA) on the network for both authentication and encryption. In addition, WPA2 can operate in an 'enterprise' mode which utilizes a centralized key server for authentication and encryption key distribution.

The use of PSK in 802.11 networks presents a vulnerability to dictionary attacks where malicious parties guess weak passphrases in order to decrypt all future packets [11]. In

addition, users can be forced off the network with deauthentication attacks and the four-way handshake used for reauthentication can be observed in order to decrypt all traffic between the AP and that particular STA. Today, these vulnerabilities can be easily exploited using hobbyist tools such as the WiFi Pineapple, an open source 802.11 penetration tester, which costs less than $100 [12]. The WiFi Pineapple incorporates a multitude of free software in order to gain access to 802.11 networks. In addition, the device is extensible with room for new attacks against standards that have not yet been written.

2.2 Current Security Solutions

The 802.11 set of standards rely on pre-shared passphrases used to encrypt and decrypt data using symmetric key cryptography algorithms. A pre-shared passphrase can be referred to as a shared secret between nodes that must be distributed before they can securely communicate. Secret pre-sharing presents core vulnerabilities in security systems from users picking weak passwords that can be easily guessed, to keys getting intercepted during the sharing process. The current major solution to the problem of having to pre-share secret keys is to use asymmetric cryptography, otherwise known as public key cryptography. In public key cryptography, a user (Alice) generates two keys in such a fashion that one key (the public key) can be used to encrypt data and only the second key (the private key) can decrypt information that was encrypted by the first key. In this type of system, Alice publishes the public encryption key for all other users to see using open channels (unencrypted communication links). This public key is then used by another user (Bob) to pass a secret message to Alice by encrypting his message using Alice's public encryption key. Bob then proceeds to send his encrypted message over an open channel to Alice. Only

Alice can decrypt the data sent by Bob as it was encrypted using the encryption key that Alice generated. These types of systems can also be used to generate a shared secret using insecure communications.

Two common algorithms enabling this type encryption are the RSA Cryptosystem [13], used to generate public and private keys, and the Diffie-Hellman Key Exchange, used to generate a shared secret over an insecure medium. The basis of these techniques is the idea that each user can generate a personal secret, perform some operation on their personal secret enabling them to transmit it to the other user, and then each user can combine their personal secret with the transmitted information to generate a shared secret that is unique and cannot be generated using the publicly transmitted information.

A potential flaw in these types of algorithms is their assumption of computational hardness. This means that the shared secrets that these algorithms are used to generate can not be computed with the publicly available information because to do so would be very difficult, so difficult that current algorithms and technology are not enough to solve the problem in a useful amount of time (i.e. it would take many, many years to figure out the secret information from the publicly available information). In the Diffie-Hellman approach, there is an assumption that it is very hard to compute certain discrete logarithms [14]. Similarly, in the RSA approach, there is an assumption of computational hardness related to the prime factorization of large integers [15]. The problem is, it is possible that these computationally hard problems may be overcome in the future through the discovery of more advanced algorithms and/or more computing power. In addition, some argue that the generally accepted assumptions of computational hardness are not necessarily the weakest assumptions that these algorithms rely on [16]. If these fundamental assumptions

are proven to be easier to solve than current knowledge indicates, then these encryption techniques that are integral to internet security would be invalidated.

More worrisome, especially in the short term, are poor implementations of these algorithms. In the mid 1990s, researches discovered that by timing how long it takes to perform certain key operations, critical cryptosystem parameters can be extracted that allow decryption of ciphertext [17]. Proposed solutions include taking action to ensure timing is somewhat random by adding extraneous information to data during encryption. Recently, it has come to light that most implementations of the Diffie-Hellman technique use a small subset of possible constant prime numbers. This creates a problem because, instead of reversing the process to gain the shared secret (which is assumed to be very hard), many possible combinations of the process can be forward calculated using existing computational resources [18]. Researchers with limited resources have demonstrated the attack on 512 bit numbers and it is likely that state sponsored agencies can perform the attack on 1024 bit numbers with current resources. This problem can be temporarily overcome by using 2048 bit numbers, but increasing the size of primes also increases encryption time; this workaround may be overcome in the future with increased computing power. Finally, all of these security algorithms rely on pseudo-random number generators to produce their private algorithm components. Any defects in the random number generation process compromises the entire security system.

While these techniques of communicating secrets over a shared medium in order to form a secure channel may have some potentially disturbing attacks, they remain strong overall when used perfectly. Some major commonalities between all of these existing systems are that they rely on external randomness through some pseudo-random number generator and

their assumptions of computational hardness. Another class of security systems, denoted Physical Layer Security, changes these underlying assumptions and sources of randomness in order to provide more options for multiple aspects system security. Since assumptions may be proven wrong, and it is always possible to have a flawed security algorithm implementation, independent security must be implemented in as many network layers as possible. Physical layer security can be used to add an independent layer of protection at the bottom of the network stack.

2.3 Physical Layer Security

PHY layer security became a topic of research in the early 1990s. The overall idea is to use the physical layer of a system to provide security. For wireless systems, this could mean utilizing directional jamming to block eavesdroppers, using tight beamforming or directional antennas to help ensure a signal only reaches the intended recipient, or hiding waveform features such as preambles in 802.11 to disrupt packet interception, among many other possibilities. This thesis will focus on the generation of encryption keys using the wireless channel between two radios.

By utilizing wireless channels, the entropy pool becomes the shared secret. This is useful since wireless channels are known to be random and unique (as described below) and there is no need for a key distribution system as radios end up with symmetric secret keys. In addition, using symmetric keys saves some time during the encrypting and decrypting processes because there is no public/private computation necessary once the initial key is generated.

In order to describe channel based encryption, a basic understanding of wireless channels

must be developed. A wireless channel is characterized as the medium through which a signal travels between the transmitter and receiver in a radio system [19]. These channels are comprised of the multiple paths in which a wireless signal simultaneously travels between the transmitter and receiver that are created by objects (or scatterers) in the environment, such as walls, people, or machinery. These paths change rapidly over time (known as channel fading) and combine in both constructive and destructive ways, influencing signal quality at the receiver. The rate of change in the channel fading is directly related to the Doppler spread of the channel, which is determined by how much physical movement is occurring, whether by scatterer movement during signal transmission, radio movement during transmission, or both. Channel fading is taken to be a random process which occurs between both ends of a radio link [19]. As two radios have access to the same channel, they share a random process which allows them to generate a shared secret.

2.3.1 Wireless Channel Model

Channel based encryption relies on fundamental properties of wireless channels. In this section, a brief overview of wireless channels is provided along with a formulation of how these channels are described for the purposes of channel based security development.

The underlying principle of channel based key generation is the idea that two nodes can form a shared secret by sampling a shared entropy source at the same time. In this situation, the wireless channel is taken as the shared source of randomness. The wireless channel is an appropriate source of entropy as it varies randomly over time and is reciprocal between two points at a given instant in time for a given frequency. This reciprocity is a fundamental property of electromagnetic wave propagation [1] and means that two nodes

can observe the state of the wireless channel and use the mutual information they gather as a shared secret. In addition, an outside observer would not be able to see the channel between the two nodes in the system as the wireless channel rapidly decorrelates in space at distances over half of a wavelength of the carrier frequency.

Typically, the wireless channel is represented as the complex transfer function, **H**, that a known signal goes through between two wireless nodes. As **H** is complex, it can be broken down as

$$H = \Re(H) + j\Im(H) = he^{j\theta} \tag{2.1}$$

Where $h = |H|$, the magnitude of the channel transfer function, and $\theta = \angle H$, the phase of the transfer function. The magnitude of the transfer function influences the amplitude of a received signal and is generally regarded to exhibit a Rayleigh distribution for typical wireless communications multipath fading channels while the phase of the transfer function influences the phase of a received signal and is generally regarded to exhibit a uniform distribution over the range $[0, 2\pi]$ [20]. Instantaneous measures of the channel are generally referred to as Channel State Information (CSI) in the frequency domain, and as a Channel Impulse Response (CIR) in the time domain.

Denoting the magnitude of **H** at time t as $h(t)$, the expression for the received signal at a radio node through a channel can be written as

$$y(t) = x(t) * h(t) + \eta(t) \tag{2.2}$$

where $x(t)$ is a transmitted signal, and $\eta(t)$ is independent noise experienced at the receiver, typically taken as Gaussian noise, and $y(t)$ is the received signal. By using a known probing signal, two nodes (denoted Alice and Bob) can estimate the channel between them by

sending the probe signal back and forth. As they cannot send the probes simultaneously, as that would require a fully duplex system in time and frequency, they must send the probes one after the other as fast as fast as possible. The channel stability over time can be characterized by the channel coherence time, T_c. The coherence time is inversely proportional to the Doppler shift in the received signal caused by motion of the radio or surrounding environment [21]. Relying on the assumption that both Bob and Alice can send and receive probe signals faster than the coherence time of their unique channel, the discrete channel estimates generated by the nodes can be written as

$$\hat{h}_{ab}[n] = h(nT) + \eta_b(nT) \tag{2.3}$$

$$\hat{h}_{ba}[n] = h(nT) + \eta_a(nT) \tag{2.4}$$

where $\hat{h}_{ab}[n]$ and $\hat{h}_{ba}[n]$ represent the estimated channel from Alice to Bob and Bob to Alice respectively, at sample n with $n = 0, 1, ..., N-1$, and $h(nT)$ represents the actual channel sampled with period T. In addition, $\eta_a(nT)$ and $\eta_b(nT)$ represent the independent noise processes observed at Alice and Bob respectively, due to noise in the receiver as well as channel variance due to forward and reverse sampling occurring non-instantaneously.

2.3.2 Wireless Channel Based Encryption Key Generation

Wireless channel based encryption key generation research began in the early 1990s with information-theoretic papers outlining the possibility of using dependent random variables to generate symmetric keys [22], [23]. In the early 2000s, fully formed, theoretical, key extraction techniques using the wireless channel were discussed [24].

Multiple key metrics are of the utmost importance to these algorithms. The first metric is Secret Bit (s-bit) generation rate, the speed at which encryption bits are produced by

the algorithm. This metric is an important consideration as it impacts how quickly keys can be generated at the beginning of a session and how quickly they can be refreshed. The second major metric is s-bit error rate, or how often corresponding bits on opposite ends of the channel fail to match. The error rate must be kept to a minimum as any one bit error means the entire key must be discarded, wasting time and resources. A third metric is mutual key information, or a measure of the advantage held by an eavesdropper. This is typically measured through examining the mutual information between key generation participants and eavesdropping observers. While important, mutual key information will not be evaluated in this thesis.

Current algorithms used for extracting keys from wireless channels differ in their methods of using the same channel state information. In [2], a method is presented and evaluated for the generation of encryption keys through the quantization of sampled channel impulse responses. This method produces a very low s-bit error rate, with a probability of error on the order of 10^{-8}. However, with the low chance of bit errors comes a low s-bit generation rate, on the order of 1 secret bit per second. In addition, the tested level-crossing algorithm requires information to be sent over an unencrypted channel in order to generate the final key. In [3], Orthogonal Frequency Division Multiplexing (OFDM) channel estimates are compared by subcarrier index, thereby comparing many narrowband channels instead of one wideband channel from each probe. While this method has a very high s-bit generation rate (on the order of 1,000 secret bits per second), and does not require the transmission of algorithm data over an unencrypted link, it has a relatively high error rate (on the order of 10^{-2}). One major commonality between these demonstrated algorithms is the use of dedicated channel probes. These probing packets require specific coordination and also take up

channel capacity. An overview of the cited algorithms can be found in section 4.1.

CHAPTER 3: WIRELESS CHANNEL SAMPLING

In order to perform channel based encryption key generation, radios must have some way to sample the wireless channel between them. Traditionally, experimental key generation implementations have employed explicit channel probes with precise timing enabling them to perform symmetric channel estimation [2], [3], [25]. Dedicated channel probes are used in sampling as they can be tightly timed in order to sample the channel while it is reciprocal. Usable reciprocity is generally maintained for a short period of time, often less than the coherence time of the channel. The coherence time is dependent on the environment and may range from tens of milliseconds to hundreds of milliseconds [26].

3.1 Effect of Sampling on Reciprocity

As it is impossible for two radios to sample a channel centered on one frequency at exactly the same time, measured reciprocity in the channel can never be perfect. Despite this, tight probe timing can allow measurements to get very close to representing the ideal reciprocity in a channel [27].

Continuing the channel model between two radios, Alice and Bob, developed in section 2.3.1, the sample timing offset in relation to the channel estimates can be expressed as

channel estimates captured at a constant offset.

$$\hat{h}_{ab}(t_a) = \hat{h}_{ba}(t_b) \tag{3.1}$$

$$t_b = t_a + \tau \tag{3.2}$$

where τ represents a constant offset between the sample times. In order to expose channel reciprocity, τ must be kept to a minimum.

For encryption key generation, channel sampling that preserves reciprocity is necessary. In order to quantize symmetric keys, deterministic channel states must be captured before they change. In addition, long term trends cannot be considered as an adversary then may be able to take advantage of channel statistics over the long term.

3.2 Application Layer Sampling Description

In this section, a framework for using communications traffic generated by the Application (APP) layer of a packet based radio system as a means of sampling the wireless channel is described. The bursty and asymmetric nature of application layer traffic presents a challenge to two radios sampling the channel symmetrically, as it cannot be assumed that each received packet directly corresponds to a packet at the other participating radio. This asymmetry is overcome through the use of an internal timer interrupt on each radio, such that there is an interrupt approximately once per channel coherence time interval. The interrupt can be viewed as a request for a packet sample. Once an interrupt occurs, the next received packet transmitted from a participating radio is used in the channel estimation process. All other packets should be disregarded for the purpose of sampling the channel, but should continue through the typical Media Access Control (MAC) pipeline. It should

be noted that if the interval between received packets on any participating station is greater than the approximate channel coherence time, the channel sampling process should be put on hold as the packets used for channel estimation may not be correlated.

Whereas in a traditional probing system there is a small, yet mostly predictable offset between symmetric channel samples, relying on APP layer traffic makes the process more stochastic. It must be understood that just because a sample is requested does not mean that a sample will be available. This changes the timing relation of equation 3.2 by adding another term, such as

$$t_b = t_a + \tau' \tag{3.3}$$

where τ' represents $\tau+\eta$ and η is a random variable representing the amount of time between a sample request and sample delivery. The amount of time between sample request and delivery is dependent on the radio system used, including how it handles radio interference and the continuous data rate of the traffic being passed. If bi-directional traffic is not expected but the system uses a packet Acknowledgment (ACK) routine, the ACKs can be used for sampling as well as data packets.

An illustration of the packet sampling mechanism is shown in Fig. 3.1. In this diagram, the received packets at a participating radio are shown over time. Note that the diagram is not to scale. In reality, the duration of a transmitted packet (on the order of microseconds) is much smaller than a sample window (on the order of milliseconds). To gather approximately symmetric samples, the process depicted in Fig. 3.1 should run on each participating receiver.

It is worth noting that in a multi-radio system, the sample request interrupt at each

receiver will not occur at the exact same time. To fix this, the timer interrupt should be roughly synchronized across the network when the process begins. Sample requests will not be at the exact same times due to imperfect synchronizing and timer interrupt variance, but they should occur within one channel coherence time window, which is fast enough to generate correlated channel estimates.

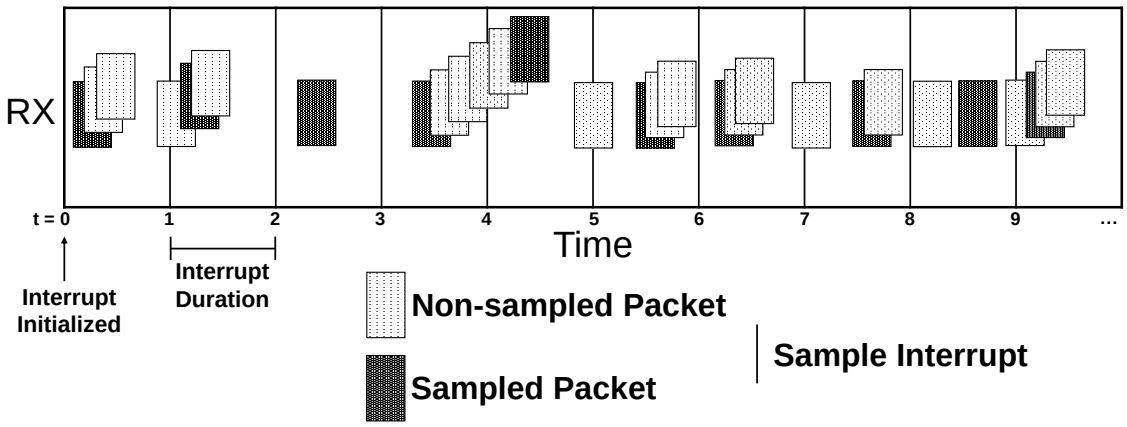

Figure 3.1: Illustration of APP layer traffic sampling using interrupts

CHAPTER 4: REAL-TIME KEY GENERATION SYSTEM

Until now, channel based encryption algorithms have been evaluated and shown to work only on experimental testbeds in controlled environments with offline processing [2], [3], [28]; they have not been put to use in real-time, standards-compliant radio networks. Presented here is the design of a real-time, standard-integrated system for channel based encryption. This system is considered to be standard-integrated (as opposed to standard-compliant). While it is not a part of any standard, it can be run within the framework of a standard, such as IEEE 802.11 provides while allowing for backwards compatibility.

4.1 Physical Layer Key Generation Algorithms

Current channel based key generation algorithms differ mainly in the way that they utilize channel estimates to produce keys. Two experimentally verified methods are described in the following sections to illustrate using channel estimate magnitudes directly versus using channel estimates to examine very short term channel trends.

4.1.1 Existing Key Generation Algorithms: CIR Level Crossing

In this section, the algorithm presented in [2] is summarized. First, a probing phase is entered where the channel between two radios is sampled using probing packets. This is accomplished through sending a known probing signal between the two nodes of interest and

comparing the received signal to the known transmitted signal. This process is illustrated in Fig. 4.1 and its output is visualized in Fig. 4.2.

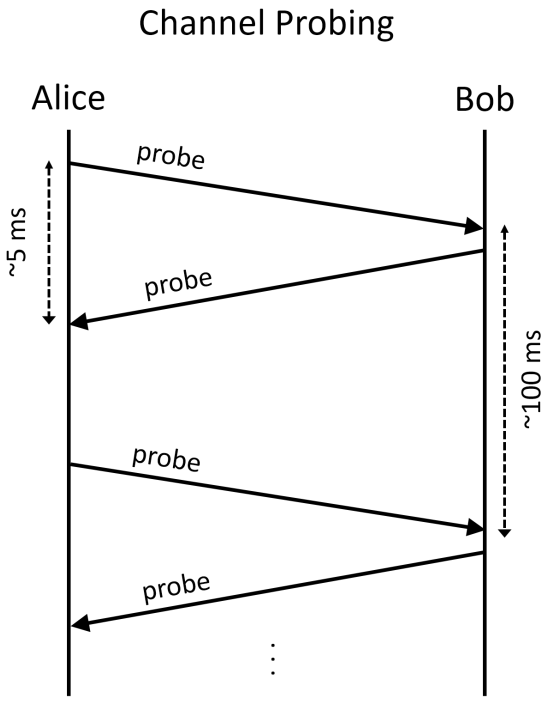

Figure 4.1: Illustration of channel probing

These probes are assembled at both ends of the link and, once enough probes are exchanged, each node independently filters the estimated channel using the maximum of the CIR to reduce the presence of large scale shadow fading. The filtered version of the channel presented in Fig. 4.2 can be seen in Fig. 4.3.

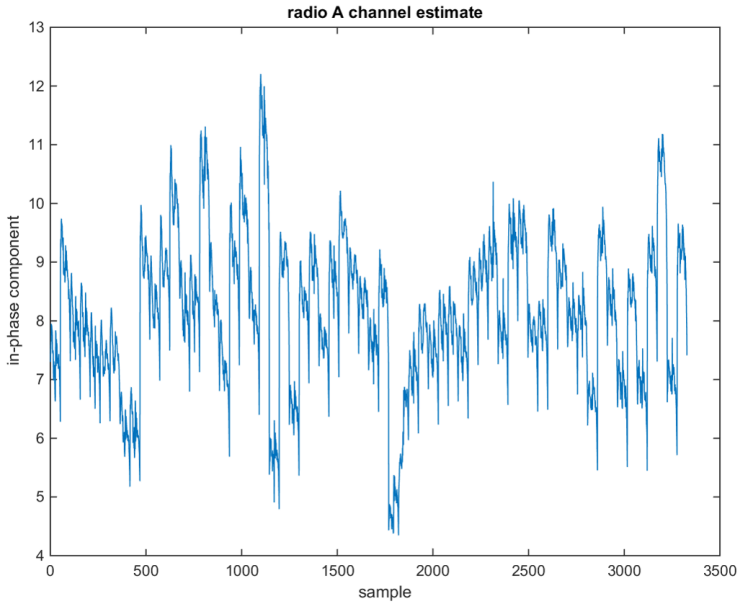

Figure 4.2: Channel estimates (peak magnitude of the CIR) at one end of the channel

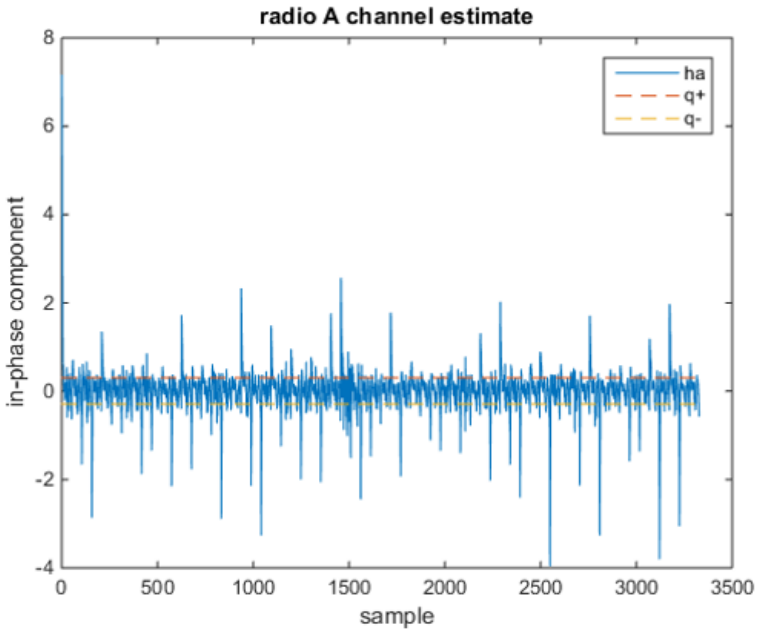

Figure 4.3: Filtered channel estimates (peak magnitude of the CIR) at one end of the channel

After filtering, the nodes compute the standard deviation of the channel to use as a

threshold to determine if the sample at each time index is a 1, 0, or should not be considered as a bit. These thresholds are referred to as q^+ and q^- for quantizing a 1 or 0, respectively. The filtered channels from each radio along with lines representing quantizer steps can be seen in Fig. 4.4.

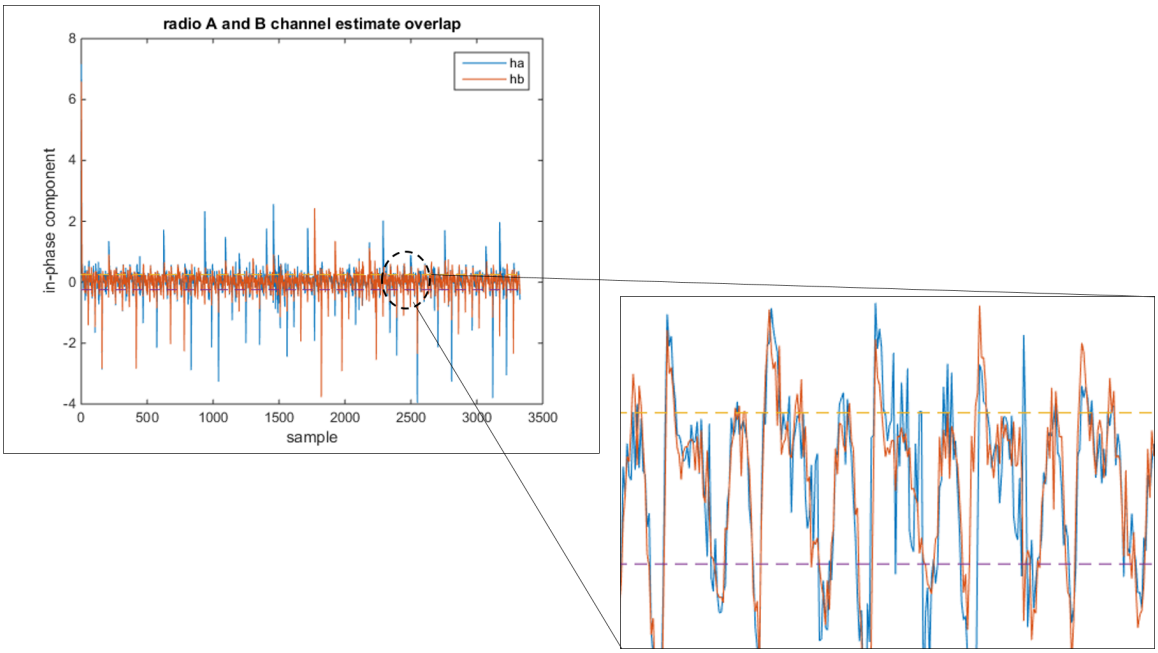

Figure 4.4: Filtered channel estimates (peak magnitude of the CIR) from both nodes overlaid with zoom in for clarity.

Next, a window is applied to the parsed bits, and a bit is considered to be present at both radios if there is a run of M same bits in a row. Runs of estimates meeting this requirement are displayed in Fig. 4.5.

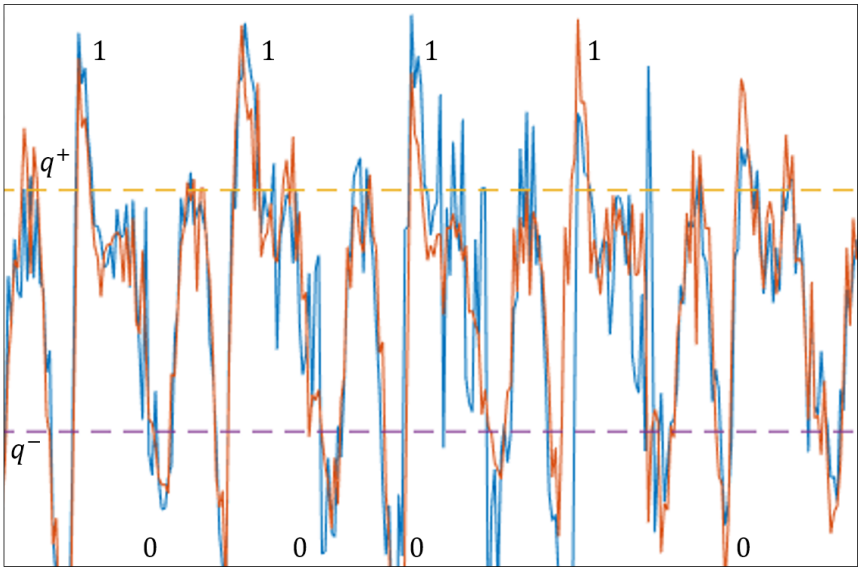

Figure 4.5: Quantized channel estimate runs on zoomed in segment from Fig. 4.4.

Finally, one radio sends the indices where it believes there is a bit to the other radio, and the other radio replies with a list of indices that it agrees contain a useful bit. Any eavesdropper would only have information about what samples are being used as bits, but not what bit the samples were parsed to due to the diversity of wireless channels. As the wireless channel is reciprocal only between the two cooperating radios due to the principle of diversity of wireless channels, the eavesdropper would not parse the same bits, therefore leaving them with a useless key.

4.1.2 Existing Key Generation Algorithms: Localized CSI Trends

In this section, the algorithm presented in [3] is summarized. Much like the previous technique, a channel probing phase is entered first. In this algorithm, the known signal is taken to be an OFDM frame such as the preambles used in the 802.11 standards. After probing, some filtering can be applied to help remove noise induced by the sampling equipment as

well as slow fading effects on the samples. Next, each probe is compared with the next probe by subcarrier, representing the CSI of the frame. This idea is illustrated between two sets of subcarrier estimates in Fig. 4.6.

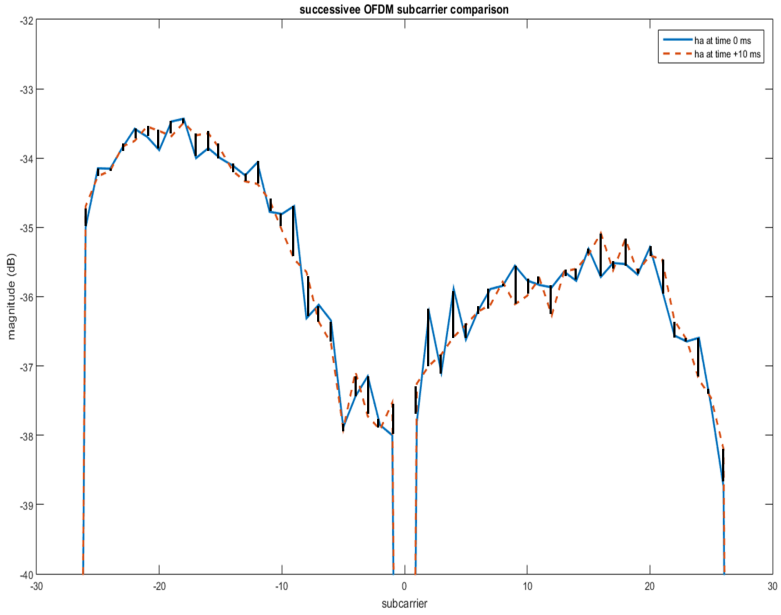

Figure 4.6: Comparison between two successive sets of subcarrier estimates (CSI).

By examining the CSI of these frames, the state of each narrow band subcarrier can be observed. Over short periods, the magnitude of the narrow band channels follow increasing or decreasing trends. These short term trends can be quantized into bits by comparing the changes between individual probes. These changes can be considered a 1 if there is an increase in magnitude, or a 0 if there is a decrease in magnitude. In order to ensure matching bits between nodes, the most common bit seen over multiple probes can be taken as the final bit. This idea is illustrated across multiple estimates of one subcarrier in Fig. 4.7.

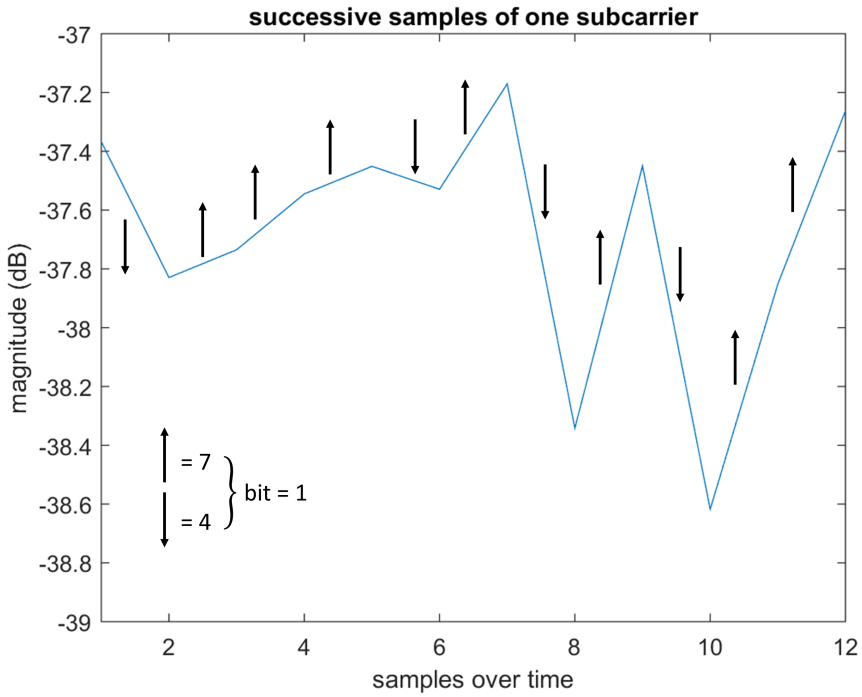

Figure 4.7: Comparison between multiple estimates (CSI) of one subcarrier.

4.2 System Description

Now that a framework for sampling APP layer traffic has been established in section 3.2, the technique can be used to integrate the key generation algorithm into the MAC layer of a radio system with minimal overhead. This integrated technique is based on the level-crossing algorithm described in section 4.1.1. This base algorithm was selected since it has been thoroughly vetted and is well known. In addition, this algorithm exhibits a low s-bit error rate, which is important for encryption reliability. First, when two radios decide to generate a key, the interrupt timer must be set on both receivers. In order to make sure the interrupts have a minimal time offset, a synchronization handshake should be introduced. This synchronization could take the form of a dedicated packet or ride on existing signaling

packets. A return packet is sent either declining to enter into key generation, or accepting while also serving as a signal to start the interrupt timer.

Next, during periods with a bidirectional stream of APP layer packets, both participating radios collect packet samples. As packet samples are collected, constant packet components, such as preambles, are used to generate channel state information. Once a predetermined number of packets are collected, the participants suspend collection to begin processing. The CSI from the collected samples is then processed following the algorithm summarized in section 4.1.1 during periods of slow traffic in order to maintain overall system performance. As the selected symmetric key generation algorithm requires bit indexes to be exchanged between nodes, a new packet may be introduced or an existing data packet can be modified to carry the algorithm information. If a different key generation algorithm is used, such as the one outlined in section 4.1.2, the use of a data exchange packet becomes unnecessary.

Once a key is established, it can be given to the APP layer to be used for high level encryption activities or, preferably, used to encrypt outgoing data within the radio stack as a supplement or replacement for existing security functions. A packet flow diagram showing how packets might flow through the MAC layer of a radio can be seen in Fig. 4.8.

The keys can be used directly and can be continually changed by generating new bits to add to the end of the existing key, but it is more likely that the keys will be used to seed a pseudo-random number generator in order to fully change the key used between every packet. Key mismatches can be determined through the sending of known packets, such as management frames. If the keys at each radio turn out to be mismatched, the process can be started over or the key can be reverted. Additionally, when a participating radio recognizes

that it is not sending or receiving enough traffic in order to maintain symmetric sampling, it should suspend the key generation process. This can be accomplished through either a management packet, or observation of traffic flows directly to keep additional overhead to a minimum.

Algorithm 1 shows a pseudocode implementation of this online key generation process. The pseudocode does not include signaling packets, but assumes that the process begins as soon as the radio is turned on.

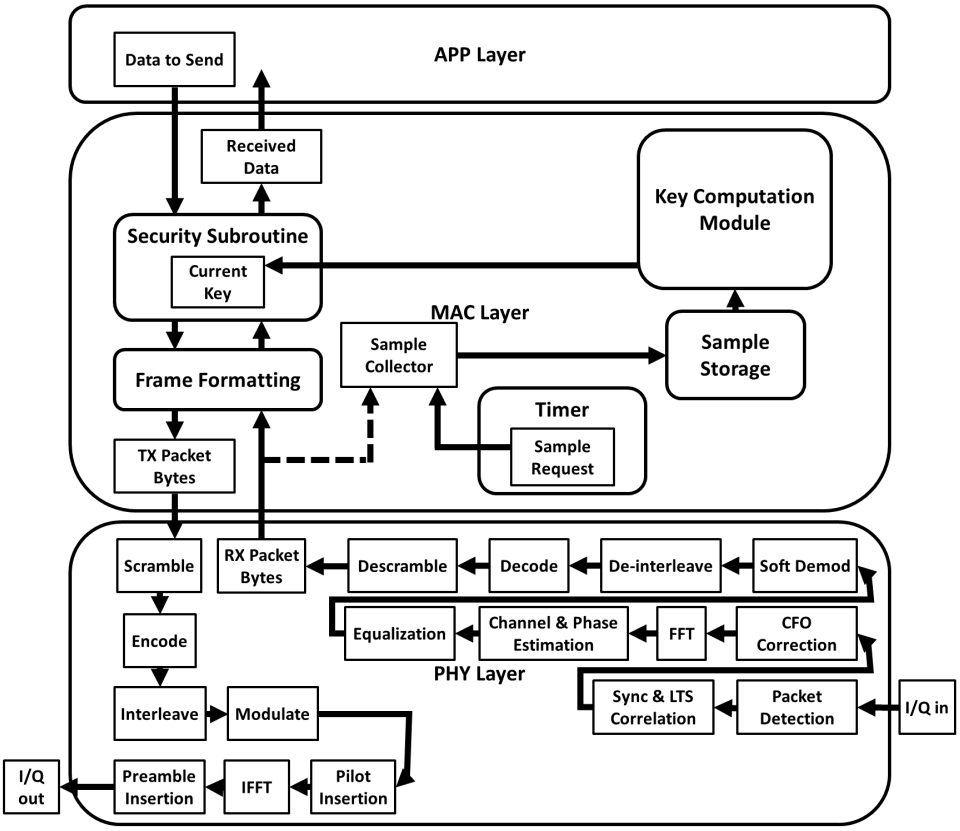

Figure 4.8: Illustration of packet flow in the real-time system

Algorithm 1 Real-Time Sampling Algorithm

```
1: global pktSample    ▷ / keep track of if a sample is needed and how many are needed
2: global numSamples    ▷ / keep track of how many samples have been collected
3: static maxNumSamples    ▷ / number of samples needed to generate a key
4:
5: procedure onSampleInterrupt
6:     pktSample++    ▷ / increment number of samples needed
7: end procedure
8:
9: procedure macHighPktRx(packet)
10:     if packetIsGood && pktSample > 0 then
11:         if pktSample > 2 then
12:             suspend sample collection
13:         else if numSamples > maxNumSamples then
14:             parse key from stored samples using level crossing algorithm
15:         else
16:             store channel sample from packet
17:             pktSample–    ▷ /update need for a new sample
18:         end if
19:     end if
20:     proceed with standard mac packet processing
21: end procedure
22:
23: procedure main
                ▷ / Initialize transmit and receive along with standard MAC processing
24:     init sampleInterrupt    ▷ / Initialize the interrupt timer
25:     while 1 do
26:                                    ▷ / Wait for packet TX or RX
27:     end while
28: end procedure
```

4.3 Case Study: Integration into IEEE 802.11

The IEEE 802.11 standard offers a widely adopted and well defined radio system for potential channel based encryption key generation. The proposed real-time key generation system could be implemented in both software with firmware and driver updates to commercial equipment and in hardware in future equipment which would provide greater speed.

In order to incorporate potentially new packet types, such as a handshake series for the start of key generation, an index exchange, and a key generation suspension packet, reserved frame subtypes can be utilized. Section 8.2.4.1.3, table 8-1 in the 802.11-2012 standard offers a list of frame identifiers, including subtypes reserved for future use [10]. Necessary key generation supporting packet types could utilize reserved space as no changes to general frame structure are needed to convey the necessary information. Using these reserved parameters without any major frame changes can be implemented with updates to existing hardware.

In addition to the extensible packet types offered by 802.11, as described in section 18.3 of the 802.11-2012 standard, a Physical Layer Convergence Procedure (PLCP) is included in every OFDM frame to facilitate frame alignment and offset synchronization and corrections. Part of the PLCP is a Long Training Symbol (LTS) as described in 802.11-2012 sections 18.3.3 and L.1.3.2 with the purpose of providing fine frequency correction as well as channel estimation. The LTS is comprised of 52 subcarriers with an alternating phase of 0° and 180° with unity magnitude and zero magnitude at the DC subcarrier.

CHAPTER 5: EXPERIMENTAL SETUP

This chapter provides details about the experimental validation that was performed to test the ideas presented in the preceding chapters. First, the experimental hardware is described in addition to the standard compliant framework that was chosen for experimentation. Next, the testing on the sampling technique that was described in section 3.2 will be detailed. Finally, the testing on the real-time system that was outlined in chapter 4 will be discussed along with preliminary statistical randomness testing. All results will be presented in chapter 6.

5.1 Experimental Setup

In order to further investigate the procedures described in chapter 4 and section 3.2, it was decided that a Software Defined Radio (SDR) would be the most versatile tool to enable implementation and rapid revision. The Mango Communications Wireless Open-Access Research Platform (WARP) v3 kit [29] was selected due to its wide usage in wireless communications research as well as its selection of proven and supported reference design for relatively quick integration.

The WARPv3 board includes Xilinx Virtex-6 Field-Programmable Gate Array (FPGA) for custom radio implementation, two transceiver chains operating in the 2.4 GHz and 5 GHz Industrial, Scientific and Medical Radio Band (ISM) with 40 MHz RF bandwidth

and 12-bit A/D and D/A chips. In addition, the radio boards include two gigabit Ethernet interfaces for application layer data transfer and radio logging and control. The radio boards contain 2 GB of DDR3 memory that is crucial for channel sample storage, which, depending on how many samples are stored at a time, can take more space than FPGA memory can provide.

In order to try these real-time designs in real-world systems, the 802.11-2012 standard was chosen as the natural test radio system. As part of their WARPv3 kit, Mango Communications offers a real-time, standards compliant, 802.11 reference design [30] which was chosen as the reference design for the implementation of APP layer sampling and real-time encryption key generation.

The WARP 802.11 reference design is a real time, FPGA based implementation of the IEEE 802.11-2012 Distributed Coordination Function (DCF) MAC layer and OFDM PHY layer. The PHY layer is implemented directly on the FPGA using Mango created IP cores. It fully supports ACK as well as request-to-send and clear-to-send functionality, along with OFDM transmitter and receiver chains as outlined in section 18 of the standard, and Direct Sequence Spread Spectrum (DSSS) receive capabilities as specified in section 16.2 of the standard to support legacy functions.

The MAC layer is implemented between two MicroBlaze softcores on the FPGA: a lower level MAC core for controlling time critical functions, such as medium access, and a higher level MAC core for less time critical applications, such as station associations and Ethernet data encapsulation. Most of the functionality specified in standard section 9.3 relating to the DCF is implemented, as well as many of the frame types specified in standard section 8.

Two major phases of evaluation were conducted and are described in the sections to follow in addition to a discussion of statistical randomness testing.

5.2 Application Layer Sampling

The first set of testing was performed with the goal of verifying that APP layer sampling could produce reciprocal channels. In order to characterize performance, the time offset between forward and reverse samples was examined, as was the variance in the offset (described using τ, τ', and η in section 3.1) in various wireless environments. In addition, visualizations of the channel in time and frequency were examined in order to verify that APP layer sampling produced a reciprocal channel.

5.2.1 Channel Estimation

As noted in section 4.3, 802.11 frames contain a PLCP which provides information for coarse and fine synchronization. In addition, the LTS component of the PLCP can be used for channel estimation.

The WARP system uses a common least-square estimator to generate channel coefficients from the LTS of received frames. As given in [31], the cost function minimized by the estimator is described as

$$S = (\mathbf{Y} - \mathbf{X}\hat{\mathbf{H}})^*(\mathbf{Y} - \mathbf{X}\hat{\mathbf{H}}) \tag{5.1}$$

where X and Y are the transmitted and received signals, respectively, $\hat{h}$ is the channel estimate, and * is the conjugate transpose operator. Thus, the least-square estimate of the channel is solved as

$$\hat{\mathbf{H}} = (\mathbf{X})^{-1}\mathbf{Y} \tag{5.2}$$

which can be viewed per subcarrier by taking a vector of complex CSI as

$$\hat{\mathbf{H}} = \left[\frac{Y_k}{X_k}\right]^T \tag{5.3}$$

with $k = 0, 1, ..., K-1$ representing each OFDM subcarrier being estimated.

5.2.2 Implementation Notes

In addition to channel estimates in the form of CSI generated as described in equation 5.3, the scheduler functionality of the 802.11 reference design was used in order to time packet requests. The fine scheduler was used for sampling purposes and had a reported timing accuracy of ± 64 μs. The local traffic generation system built into the design was used to generate APP layer data packets to send between participating nodes. Finally, the WARPnet Python experimental framework was used to control experiments as well as capture logs including sampled packets along with their channel estimates for offline analysis. The experimental framework provides a low overhead method of capturing system information about all radios in the network.

Primary testing was carried out between two nodes, an AP, Alice, and an STA, Bob. Channel samples were taken using the system described in section 3.2, and all data was sent to a central host computer for processing. In order to test the sampling technique with various multipath and mobility conditions, a Spirent SR5500 wireless channel emulator was used. The channel emulator setup can be seen in Fig. 5.1. In addition, the radios were setup across a lab consisting mostly of office space, as depicted in Fig. 5.2. Results are discussed in section 6.1.

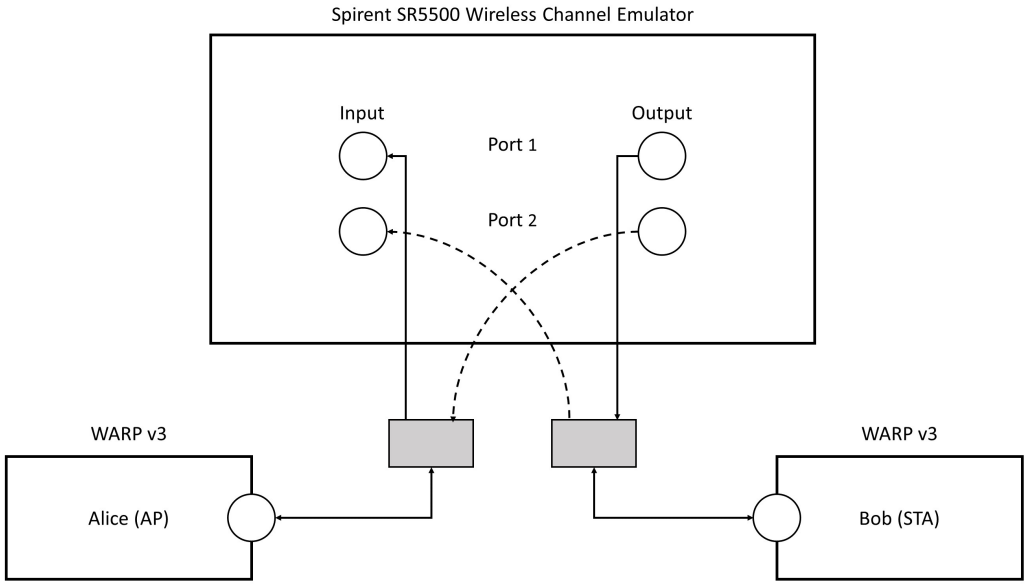

Figure 5.1: Channel emulator experimental configuration

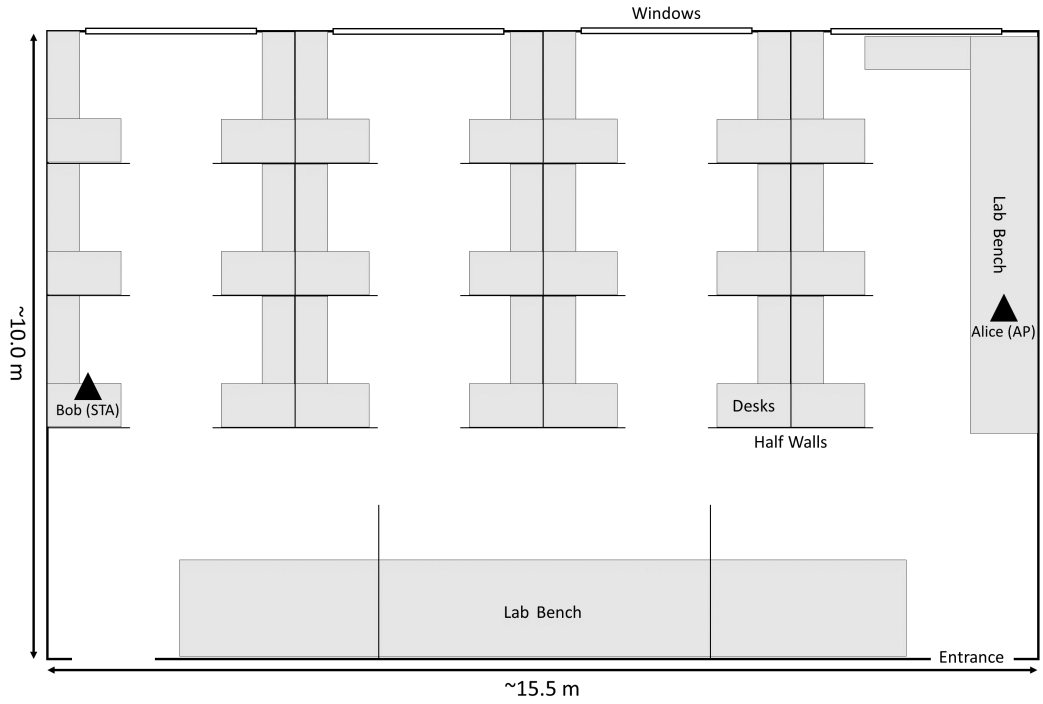

Figure 5.2: Over the air experimental configuration

5.3 Real-Time Encryption Key Generation

The second set of testing was performed with the goal of verifying that symmetric encryption key generation could occur in real time while maintaining backwards compliance with standard systems. In order to characterize performance, the s-bit generation rate and error rate were observed. Additionally, added latency was noted and standards compliance was verified.

Primary testing was carried out using two nodes, an AP and an STA. Channel samples were taken using the system described in section 3.2, but for this experiment all processing was performed on the WARP hardware, either on the FPGA directly or on a MicroBlaze softcore. The key generation algorithm discussed in section 4 was implemented in the high level mac with channel estimates stored on the DDR3 memory while waiting for processing.

APP layer traffic was generated using both ping commands and file transfers between host PCs attached to the AP and STA. In order to facilitate sampling, APP layer data was sent at a rate such that packets were sent much faster than once per sample window, nominally 2 Mbps. Two nodes were set up approximately twenty feet apart in a research lab with mostly office space. Experimentation was carried out over the air on both congested as well as unoccupied channels. Standard 2.4 GHz monopole antennas were used so as to resemble a typical consumer wireless network. For the purpose of this test, the sampling period used was 80 ms as the environment had some motion due to human foot traffic, but there were no very rapidly changing components. Nodes were spaced in a similar configuration to that depicted in Fig. 5.2.

Keys were generated using a window over the channel state information derived from

the preamble of the 802.11 data packets. A total of 80 packets were considered at a time. This grouping allowed full keys to be generated, even if a bit error occurred during one segment of the process. If one segment of the key was discovered to be mismatched through the reception of a non-decodable test packet, the key was reverted by one segment and the process continued. In addition to practicality, this operation mirrors the possible use case of changing a key over time. Success was measured by bit error rate as well as raw bit generation speed.

Standards compliance was spot-checked though the use of a commercial WiFi device. A mobile phone was attached to the modified AP and the internet was accessed in various configurations. Spot-checking occurred while key generation was in progress with the modified STA and while the modified STA was not connected to the modified AP.

5.4 Key Randomness

As the intended use of the keys generated using this system is data encryption, it is imperative that the keys are sufficiently random. Typically, keys generated using wireless channels are considered theoretically random as the underlying processes that the keys are parsed from, such as the traditional Rayleigh fading channel, are known to be random [22], [23], [32].

Concerns about non-randomness stem from the algorithms used to parse bits from channel samples. These algorithms may unintentionally (or intentionally) introduce defects in the keys. Confidence in key randomness is commonly gained by passing large amounts of generated bits through a battery of statistical tests to search for defects such as frequent runs of the same bit, repeating patterns in the bits, and non-uniform distributions of bit

values. Here, a tool published by the National Institute of Standards and Technology [33] was used to check for statistical defects as it is considered the industry standard test suite. This method was chosen as it was used for the original randomness verification of the algorithm chosen for the real-time system [2]. As suggested in [34], six tests were chosen: block frequency, longest run of ones in a block, spectral, non-overlapping template matching, serial, and cumulative sums. These tests were chosen as they focus on the distribution of ones and zeros within the key and the detection of pattern defects, but are different from the tests chosen in [2] because there were many more bits available for statistical testing in that work. A detailed explanation of the tests can be found in [33].

As a large number of bits are needed to check for statistical defects, keys used for this testing were generated using channel samples collected by the APP layer sampling system while the keys were generated from the samples using offline processing to maximize key length and minimize error rate. This was necessary as approximately 10^6 bits were needed for basic analysis so bit generation had to occur as fast as possible.

CHAPTER 6: EXPERIMENTAL RESULTS AND DISCUSSION

The experiments outlined in chapter 5 were carried out and results are discussed here. First, the described sampling technique is characterized in order to gain confidence that reciprocal channels can be generated. Next, the real-time system implementation is tested in order to determine if it is a viable path forward for future experimentation and standards integration for physical layer key generation. Finally, key randomness is investigated to ensure that the keys generated using this system are useful for encryption purposes.

6.1 Application Layer Sampling

Ten tests were conducted to evaluate the effects of multiple paths, low levels of Doppler shift, and packet generation rate on the reliability of APP layer sampling. The final test was performed over the air in an office environment. The nodes were approximately forty feet apart and did not have line of sight between their antennas. Samples were collected using an unoccupied wireless channel. The setup parameters of each test are described in table 6.1. In each case, the sample period was set at 10 ms and statistics are calculated from 48,000 sampled packets captured over four trials per test parameter set.

The first metric of interest is how consistently samples were grabbed at the requested interval. A comparison of the sampling interval for each of the parameter sets can be seen in table 6.2.

Table 6.1: Summary of test parameters used to evaluate APP layer sampling system

Test Case	Channel	Num Paths	Doppler (Hz)	Packet Gen. Rate (ms)
1	static	1	-	5.0
2	static	1	-	2.0
3	Rayleigh	1	4.25	5.0
4	Rayleigh	1	4.25	2.0
5	Rayleigh	1	8.27	5.0
6	Rayleigh	1	8.27	2.0
7	Rayleigh	3	4.25	5.0
8	Rayleigh	3	4.25	2.0
9	Over the Air	-	-	5.0
10	Over the Air	-	-	2.0

Table 6.2: Comparison of estimate period for each test parameter set

Test Case	1	2	3	4	5
Mean Est Period (ms)	9.98	10.01	9.99	10.01	9.99
Std. Dev. (ms)	0.38	5.46	0.38	5.71	0.52
Test Case	6	7	8	9	10
Mean Est Period (ms)	10.01	9.99	10.01	10.01	10.07
Std. Dev. (ms)	5.20	0.47	5.65	2.99	10.80

Right away, it is apparent that using an APP layer packet generation rate of 2.0 ms does not work well in this implementation. This is most likely due to a combination of too much strain on the 802.11 reference design along with problems with one radio becoming dominant in the channel by flooding it with packets before the other could respond.

While the deviation in the sampling period does seem to increase as expected for higher levels of Doppler and high orders of multipath, all sets of parameters have an average sampling period of 10 ms, which was the requested rate. Over the air test ten (2.0 ms packet generation rate) can be considered encouraging as the average sample period was close to 10 ms and the majority of the samples were within 20 ms, which is well within one channel coherence window for the test case. In the case of a 5.0 ms packet generation rate (over the air test nine), the deviation in sample period is minimal, ensuring the estimates are within one coherence window.

The second major measure of interest is the delay between forward and reverse channel estimation, represented by τ' in equation 3.3. A comparison of the sampling delay for each of the parameter sets can be found in table 6.3.

Table 6.3: Comparison of sampling delay between the forward and reverse samples

Test Case	1	2	3	4	5
Mean Delay (ms)	11.69	250.69	13.61	202.30	14.57
Std. Dev. (ms)	1.51	32.82	3.24	59.72	2.44
Test Case	6	7	8	9	10
Mean Delay (ms)	196.85	12.61	171.90	8.28	13.11
Std. Dev. (ms)	51.44	2.28	30.66	5.90	41.27

Again, the trials conducted with a packet generation period of 2.0 ms exhibit unexpected behavior. As they show such a high sampling offset, well over a channel coherence window, they would not be useful for observing reciprocity in the channel. The large offset may confirm that the packets may be colliding at a higher than usual rate, or it may indicate

that the hardware cannot support such a fast bi-directional packet stream.

On the contrary, the trials conducted with a packet generation period of 5.0 ms performed as expected with each one averaging an offset of less than 15.0 ms. The air channel performance was reasonable, with the high deviation most likely caused by packet loss due to weak signal strength as a result of the non line of sight setup as opposed to transceiver overloading. This performance is fast enough to observe reciprocity for slowly varying channels, i.e. channels seen in an office environment but not necessarily channels seen in vehicular communications.

In order to gain confidence in the capture of channel reciprocity using the APP layer technique, CSI representing the channel coefficients for each utilized subcarrier in the 802.11 LTS along with phase estimates were viewed for a selection of sample pairs. An example can be seen in Fig. 6.1.

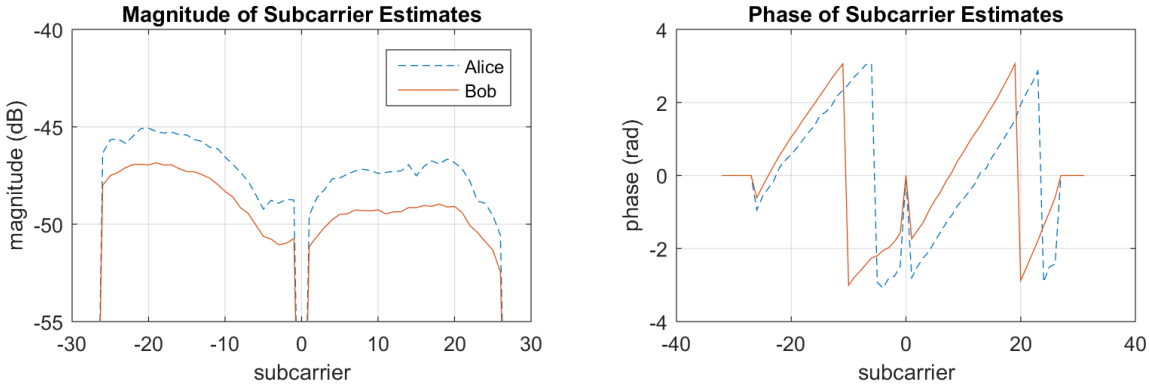

Figure 6.1: Comparison of subcarrier estimates between forward and reverse samples

As can be seen in the magnitude plot, the subcarrier magnitudes follow the same trend at both ends of the channel. This helps demonstrate that the channels are sampled so as to preserve reciprocity. It is acceptable that the magnitudes at each end are different by a scalar value as the presented physical layer key generation algorithms are interested in short

term channel trends, not absolute values. The magnitudes will almost always have some offset due to transceiver differences unless some explicit calibration procedure is undertaken. While not as close as the magnitudes, the channel phase estimates are acceptable as they follow a similar rate of change over the subcarriers. The actual value of the phase will almost never be the same because noise added by the transceivers through frequency mismatches add offsets to the phase. In addition, the phase of a channel is generally accepted to vary faster than the magnitude of the channel [1].

Finally, the time domain CIR peak magnitude was compared at both radios by overlaying the two sets of estimates. A short window of the overlay can be found in Fig. 6.2.

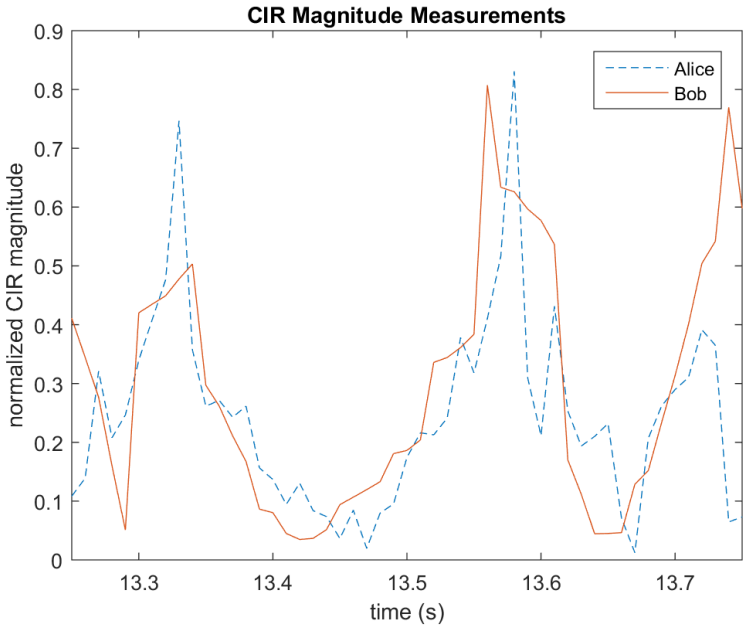

Figure 6.2: Comparison of CIR peak magnitude estimates for forward and reverse samples

It is apparent that the estimates follow the same general trend, even without any corrective filtering applied. This is significant as it confirms that the APP layer sampling technique can capture reciprocity in the wireless channel which is necessary in order to per-

form symmetric key generation. A final indication of reciprocity comes from the correlation of the two sets of channel estimates, which for these over the air samples is represented as a Pearson correlation coefficient of 0.57 which is generally accepted to represent a correlation in the data.

6.2 Real-Time Encryption Key Generation

Real-time system testing results are summarized in Table 6.4 as a benchmark of first generation system performance.

Table 6.4: Summary of experimental results using modified WARP system

Experiment duration	20 min
Interrupt timer	80 msec
Average s-bit rate	0.63 s-bits/sec
Average bit-error rate	6.7%

Overall, the channel-based key generator added into the WARP 802.11 reference implementation generated symmetric bits with a secret bit rate of 0.63 s-bits/sec and with an average bit error rate of 6.7%. While this error rate seems low, a single bit mismatch renders the entire key segment useless. As a result, the overall segment mismatch rate was close to 33.3% with the average segment containing five bits generated from 80 packet samples. This statistic means that for a complete key of 128 bits, 35 key segments had to be generated on average with 10 of the segments discarded due to the presence of a bit error. While the bit generation rate of this system is lower than research implementations [35], [25], this may be acceptable as this algorithm can continually run without introducing

the overhead required for dedicated probing packets, so long as sufficient APP layer traffic is moving across the network.

Some processing overhead was introduced to the WARP 802.11 MAC as a result of running the algorithm. Round-trip latency for APP layer traffic was increased from an average of 0.85 ms to 1.21 ms, representing a 42.4% increase. This increase can be largely attributed to the processing necessary to extract keys from the channel state information. The algorithm was run on the same MicroBlaze processing core used for MAC high functions on the WARP 802.11 reference as opposed to an isolated processing environment. No change in effective data throughput was observed after the introduction of the key extraction algorithm to the WARP 802.11 reference.

In the current implementation, this technique seems to be best suited for augmenting WPA2 or similar processes. A sample use case could involve starting with a passphrase and then using this algorithm to change the key over time in a random fashion. This process can be equated to using the WPA2 passphrase as a means of authentication and initial encryption, and then shifting the encryption responsibilities over to the physical layer security algorithm. This technique also has potential to augment standards like HTTPS in unsecured networks. Even in places where the 802.11 network is unsecured, this method can be used to secure individual user connections and can generate enough bits to make keys from scratch. A slower key generation rate could be traded for lower bit error rate to ensure that fully symmetric keys are generated on the first try.

6.3 Key Randomness

Key randomness statistical test results can be found in Table 6.5. These results were generated using 10^6 bits and are reported as a P-value where $P > 0.01$ indicates sequence randomness. The P-values are calculated by the test suite differently for each test and are a measure of confidence that the sequence is random based on the test results. Channel data from APP layer sampling test scenario nine was used to generate symmetric keys using offline processing. Test scenario nine was chosen as it showed the best over the air results and offline processing was used in order to maximize s-bit generation rate.

Table 6.5: Results of randomness tests on generated bits

Test	P-value
Longest Run	0.037590
Frequency (block)	0.867725
Discrete Fourier	0.653027
Non-overlapping Template	0.884280
Serial	0.961719
Cumulative Sums	0.123029

These results provide the impetus to continue development of this real-time system into a mature commercial implementation. While the results are encouraging because each selected test passed, they are only the start of randomness testing as the tests must be repeated using millions of bits in various environments in order to gain confidence that the keys are sufficiently random. Future systems must be reevaluated for randomness as implementation decisions may add an unintentional bias to the keys. All real-time key generation

algorithm implementations should be tested for statistical defects. More complete testing may be acomplished by following the procedure outlined in [34].

CHAPTER 7: CONCLUSIONS

The purpose of this work was to help begin the transition of physical layer security techniques from theoretical constructs and purpose-built systems to widely accepted standards and commercial equipment. Specifically, this work focused on the generation of encryption keys using the physical wireless channel between two radios. The major challenges faced were: 1. how to sample the wireless channel in a manner retaining reciprocity without introducing heavy overhead and 2. how to accomplish generating encryption keys from the wireless channel between two radios within the framework provided by existing standards. These hurdles were overcome through the establishment of a method to sample the wireless channel using data generated at the application layer. This technique was then incorporated into a real-time encryption key generation system design based on the widely adopted IEEE 802.11 standard. Finally, initial experimental validation was performed on these systems to establish their viability for continued development.

Through experimental verification, it was demonstrated that despite the timing variance incurred as a result of using APP layer data traffic to estimate wireless channels, estimate reciprocity can be achieved. Further, the reciprocity is enough to generate symmetric encryption keys in a real-time processing system implemented on a SDR running a standards compliant radio MAC and PHY design. Encouraging results from statistical testing provided confidence that the keys generated through these systems are indeed random.

7.1 Recommendations for Future Work

While this system has shown that it is possible to generate PHY layer encryption keys in a standards compliant environment using application layer traffic, more work has to be performed to determine the optimal sampling speed and algorithm confidence parameters. This should include a characterization of how various delay spreads and scattering environments influence key randomness to ensure cryptographic soundness. Real-time channel coherence time estimation, similar to what is discussed in [36], should be investigated in order to dynamically alter algorithm parameters to ensure key randomness and generation rates. Key mutual information between active key generation participants and eavesdroppers must be evaluated to understand the advantage held by an eavesdropper. Finally, future system testing will need to be conducted in a variety of measured and emulated environments to quantify key error rates and randomness in different situations.

LIST OF REFERENCES

[1] T. Rappaport, *Wireless Communications: Principles and Practice*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2nd ed., 2001. 2, 10, 42

[2] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, "Radio-telepathy: Extracting a Secret Key from an Unauthenticated Wireless Channel," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, MobiCom '08, (New York, NY, USA), pp. 128–139, ACM, 2008. 2, 13, 15, 19, 37

[3] C. Sahin, B. Katz, and K. R. Dandekar, "Secure and Robust Symmetric Key Generation using Physical Layer Techniques under Various Wireless Environments," in *Radio and Wireless Symposium (RWS), 2016 IEEE*, Jan 2016. 2, 13, 15, 19, 23

[4] B. Z. Katz, C. Sahin, and K. R. Dandekar, "Real-Time Wireless Physical Layer Encryption," in *IEEE Wireless and Microwave Technology Conference (WAMICON 2016)*, April 2016. 2

[5] "IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-1997*, pp. i–445, 1997. 4

[6] S. R. Fluhrer, I. Mantin, and A. Shamir, "Weaknesses in the Key Scheduling Algorithm of RC4," in *Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography*, SAC '01, (London, UK, UK), pp. 1–24, Springer-Verlag, 2001. 4

[7] "IEEE Standard for information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 6: Medium Access Control (MAC) Security Enhancements," *IEEE Std 802.11i-2004*, pp. 1–190, July 2004. 5

[8] E. Tews and M. Beck, "Practical Attacks Against WEP and WPA," in *Proceedings of the Second ACM Conference on Wireless Network Security*, WiSec '09, (New York, NY, USA), pp. 79–86, ACM, 2009. 5

[9] A. Lodhi, "Cryptanalysis of IEEE 802.11i TKIP," Norwegin University of Science and Technology, July 2010. 5

[10] "IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp. 1–2793, March 2012. 5, 29

[11] A. Tsitroulis, D. Lampoudis, and E. Tsekleves, "Exposing WPA2 Security Protocol Vulnerabilities," *Int. J. Inf. Comput. Secur.*, vol. 6, pp. 93–107, Mar. 2014. 5

[12] "WiFi Pineapple." `https://www.wifipineapple.com)`. 6

[13] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Commun. ACM*, vol. 21, pp. 120–126, Feb. 1978. 7

[14] J. Brown-Cohen, "Evidence that the Diffie-Hellman Problem is as Hard as Computing Discrete Logs," 7

[15] R. L. Rivest and B. Kaliski, "RSA Problem," Dec. 2003. 7

[16] D. Boneh and R. Venkatesan, "Breaking RSA May Be Easier Than Factoring (Extended Abstract)." 7

[17] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '96, (London, UK, UK), pp. 104–113, Springer-Verlag, 1996. 8

[18] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, B. VanderSloot, E. Wustrow, S. Zanella-Béguelin, and P. Zimmermann, "Imperfect forward secrecy: How diffie-hellman fails in practice," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, (New York, NY, USA), pp. 5–17, ACM, 2015. 8

[19] A. Molisch, *Wireless Communications*. Wiley-IEEE Press, 2005. 10

[20] A. Sayeed and A. Perrig, "Secure wireless communications: Secret keys through multipath," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3013–3016, March 2008. 11

[21] W. C. Jakes and D. C. Cox, eds., *Microwave Mobile Communications*. Wiley-IEEE Press, 1994. 12

[22] U. M. Maurer, "Secret key agreement by public discussion from common information," *IEEE Transactions on Information Theory*, vol. 39, pp. 733–742, May 1993. 12, 36

[23] R. Ahlswede and I. Csiszar, "Common randomness in information theory and cryptography - Part I: Secret sharing," *IEEE Transactions on Information Theory*, vol. 39, pp. 1121–1132, Jul 1993. 12, 36

[24] H. Koorapaty, A. Hassan, and S. Chennakeshu, "Secure information transmission for mobile radio," in *Information Theory, 1998. Proceedings. 1998 IEEE International Symposium on*, pp. 381–, Aug 1998. 12

[25] M. Wilhelm, I. Martinovic, and J. Schmitt, "On key agreement in wireless sensor networks based on radio transmission properties," in *Secure Network Protocols, 2009. NPSec 2009. 5th IEEE Workshop on*, pp. 37–42, Oct 2009. 15, 43

[26] H. MacLeod, C. Loadman, and Z. Chen, "Experimental studies of the 2.4-GHz ISM wireless indoor channel," in *Communication Networks and Services Research Conference, 2005. Proceedings of the 3rd Annual*, pp. 63–68, May 2005. 15

[27] S. Haile, "Investigation of Channel Reciprocity for OFDM TDD Systems," Master's thesis, University of Waterloo, Waterloo, Ontario, Canada, 2009. 15

[28] A. J. Pierrot, R. A. Chou, and M. R. Bloch, "Experimental aspects of secret key generation in indoor wireless environments," in *2013 IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 669–673, June 2013. 19

[29] "WARP project." `http://warpproject.org`. 30

[30] "Mango Communications 802.11 Reference Design." `http://mangocomm.com/802.11`. 31

[31] W. G. Jeon, K. H. Paik, and Y. S. Cho, "An efficient channel estimation technique for OFDM systems with transmitter diversity," in *Personal, Indoor and Mobile Radio Communications, 2000. PIMRC 2000. The 11th IEEE International Symposium on*, vol. 2, pp. 1246–1250 vol.2, 2000. 32

[32] J. Cardinal and G. V. Assche, "Construction of a shared secret key using continuous variables," in *Information Theory Workshop, 2003. Proceedings. 2003 IEEE*, pp. 135–138, March 2003. 36

[33] A. Rukhin, J. Soto, J. Nechvatal, E. Barker, S. Leigh, M. Levenson, D. Banks, A. Heckert, J. Dray, S. Vo, A. Rukhin, J. Soto, M. Smid, S. Leigh, M. Vangel, A. Heckert, J. Dray, and L. E. B. Iii, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," 2001. 37

[34] C. Kenny, "Random Number Generators: An Evaluation and Comparison of Random.org and Some Commonly Used Generators," Master's thesis, Trinity College Dublin, Dublin, Repulbic of Ireland, 2005. 37, 46

[35] S. Jana, S. N. Premnath, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy, "On the Effectiveness of Secret Key Extraction from Wireless Signal Strength in Real Environments," in *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*, MobiCom '09, (New York, NY, USA), pp. 321–332, ACM, 2009. 43

[36] T. Yucek, R. Tannious, and H. Arslan, "Doppler spread estimation for wireless OFDM systems," in *Advances in Wired and Wireless Communication, 2005 IEEE/Sarnoff Symposium on*, pp. 233–236, April 2005. 48

LIST OF ACRONYMS

WEP Wired Equivalent Privacy

IV Initialization Vector

WPA WiFi Protected Access

WPA2 WiFi Protected Access Second Generation

RC4 Rivest Cipher 4

AES Advanced Encryption Standard

TKIP Temporal Key Integrity Protocol

PSK Pre-Shared Key

AP Access Point

STA Station

PHY Physical

APP Application

CSI Channel State Information

CIR Channel Impulse Response

s-bit Secret Bit - Usually referring to bits generated for encryption usage

OFDM Orthogonal Frequency Division Multiplexing

DSSS Direct Sequence Spread Spectrum

APP Application - Usually referring to the Application Layer of a system

PHY Physical - Usually referring to the Physical Layer of a system

MAC Media Access Control

ACK Acknowledgment

PLCP Physical Layer Convergence Procedure

LTS Long Training Symbol

WARP Wireless Open-Access Research Platform

SDR Software Defined Radio

ISM Industrial, Scientific and Medical Radio Band

FPGA Field-Programmable Gate Array

DCF Distributed Coordination Function - Usually referring to the MAC scheme of the IEEE 802.11 standard

LIST OF VARIABLES

$\mathbf{H}$ Complex transfer function representing a wireless channel

t Continuous time

h Magnitude of $\mathbf{H}$ at time t

θ Phase of $\mathbf{H}$ at time t

η Random noise process, typically Gaussian

x Transmitted signal

y Received signal

$\hat{h}$ Estimated magnitude of $\mathbf{H}$

n Discrete time sample

N Total number of samples

T Sample period for discretization

τ Constant time offset between forward and reverse channel samples

τ' Random time offset between forward nd reverse channel samples

q Bit quantization threshold

M Run of the same bits

$\mathbf{Y}$ Frequency domain received signal

$\mathbf{X}$ Frequency domain transmitted signal

$\hat{\mathbf{H}}$ Frequency domain channel estimate

k OFDM Subcarrier index

K Total number of subcarriers

Y_k Received subcarrier k

X_k Transmitted subcarrier k